TRIBUS: An end-to-end automatic speech recognition system for European Portuguese

Carlos Carvalho[1]

[12]Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal
[12]INESC-ID, Lisbon, Portugal

carlos.f.carvalho@tecnico.ulisboa.pt

Abstract

[1] End-to-end automatic speech recognition (ASR) approaches have emerged as a competitive alternative to traditional HMM-based ASR systems. Unfortunately, most end-to-end ASR systems are not easily reproduced since they require vast amounts of data and computational resources that are only available for a reduced set of companies and labs worldwide. Consequently, the performance of these systems is not very well known for low resource languages to the best of our knowledge. European Portuguese is one of those languages. In this work, we present a set of experiments to train and assess some of the most current successful end-to-end ASR approaches for European Portuguese. The proposed system, named TRIBUS, is a hybrid CTC-attention end-to-end ASR combining data from three different domains: read speech, broadcast news and telephone speech. For comparison purposes, we also train a state-of-the-art HMM-based baseline on the same data. Experimental results show that TRIBUS achieves 8.40% character error rate (CER) on the broadcast news test set without the need of a language model, which is comparable to the strong baseline result, 4.33% CER, on the same set using an in-domain language model. We consider this result quite promising, especially for highly unpredictable vocabulary ASR applications. Finally, and more notably, a novel way of training CTC-based models using a memory-based approach, that performs better than only using CTC alone, was developed.

Index Terms: automatic speech recognition, end-to-end, hybrid CTC-attention, low resources, memory-based approaches

1. Introduction

Speech recognition technology is submerged in our society more than ever. Products like Siri, Cortana, Google Now and Amazon Echo Alexa which belong to big companies, like Apple, Microsoft, Google and Amazon, respectively, are part of our every day lives. This high tech translates into a significant number of applications (e.g., healthcare and autonomous vehicles) which have contributed to increase the quality of live in our society.

Traditionally, large vocabulary continuous speech recognition (LVCSR) systems rely on sophisticated modules including acoustic, phonetic and language models, which are manually created by specialized computational linguists and engineers. Since all these modules do not optimize the same goal, the ASR system final objective will have more difficulties in achieving the global optimum. Furthermore, the Hidden Markov models (HMMs) systems and n-gram language models used make conditional independence assumptions, whereas real speech does not follow those strict assumptions. To overcome these limitations, by replacing an HMM-based system with a deep neural network, one gets a new model that trains with a global optimization procedure. Also, by removing the engineering required for the usual alignment, bootstrapping, clustering and decoding with finite-state transducers (FSTs), characteristic of the HMM-based models, the training and decoding process becomes more straightforward. This new technology, named end-to-end, directly maps an input sequence of acoustic features to an output sequence of tokens, i.e., characters, sub-words or words.

Some widely used contemporary end-to-end approaches are: connectionist temporal classification (CTC) [1, 2], attention encoder-decoder (AED) [3, 4] and RNN Transducer (RNN-T) [5]. CTC main problem is that it is not capable of modelling language [6] because it considers each label in the output sequence to be independent of each other. To solve CTC-based models independence assumption, Alex Graves developed the RNN-T system. As opposite to CTC, RNN-T does not make assumptions about label independence when enumerating the hard alignments. However, the main disadvantage of CTC-based and RNN-T systems is that, since they first enumerate all hard alignments and then aggregate them, there could be many illogical paths. Attention-based models solve this problem by creating a direct soft alignment between input and output, with the support of an attention mechanism. One of the main issues of attention-based models is the monotonic alignment problem. As a result, the attention mechanism can allow extremely non-sequential alignments between input frames and output tokens [7]. To solve this, hybrid CTC-attention models were proposed in [8]. These models use the advantages of both CTC-based and attention-based architectures in training and decoding.

The main drawback of these end-to-end systems, mentioned above, is that they require a considerable number of training hours to achieve state-of-the-art performance results when compared to traditional HMM-based systems [9]. For English ASR, corpora such as TED-LIUM [10], and Librispeech [11] offer great possibilities for researchers to experiment and compare large end-to-end ASR systems. However, this is not the case for European Portuguese, mainly due to the lack of large scale speech data resources publicly available, either paid or for free. As a result, all corpora used for the experiments of this work is from INESC-ID.

The main contribution of this work is the creation of the first known end-to-end ASR system for European Portuguese, by using one of the most successful end-to-end ASR approaches, for a low resource scenario. We also aim to investigate and propose novel ways for either creating improved speaker-invariant representations or incorporating memory modules into current architectures that can improve the current state-of-the-art.

The remainder of this document is organized as follows. We

[1]This extended abstract is an extension of a paper that was submitted to IberSPEECH 2020 conference.

start by presenting some related work in Section 2. Next, we describe the main corpus used to train the end-to-end system and the strong HMM-based baseline in Section 3. Section 4 gives a brief description of the acoustic feature extraction and describes the main architecture used to train the end-to-end ASR TRIBUS system, a starting point for all further mentioned experiments. In Section 5 we will mention the creation and setup for the speaker invariant systems. Section 6 will describe the novel system that uses a memory-based system with CTC for ASR. Section 7 will detail some experimental setup, including the results for each experiment of this work. At last, a concluding summary and future work are presented in Section 8 and Section 9, respectively, and in Section 10 we mention acknowledgements.

2. Related work

Since 1980, traditional speech recognition systems were composed by HMMs, which model the probability of going from one acoustic state (generally a triphone) to another, and Gaussian mixture models (GMMs) that model the probability of occurrence of that particular acoustic state.

The combination of the HMMs and GMMs constitute what is commonly known as the acoustic model of the speech recognition system, which is eventually combined with a language model and a pronunciation dictionary to then generate the text transcriptions, through a decoding process. The creation of these modules for HMM-GMM systems is favourable when few data is available to train since they already contain some speech and language knowledge. Nonetheless, the creation of these individual and complex systems can limit the potential performance of speech recognition, mainly because they can be poor approximation models of reality, e.g., forcing an algorithm to use a phonetic representation can limit the speech system's performance [12].

In 2006, artificial neural networks (ANNs) resurged with a new name: deep learning. This occurred mainly because it started to be possible to train networks with more layers, using a new greedy layer-wise unsupervised pretraining technique [13]. Today, we know that unsupervised pretraining is not required to train deep neural networks as a result of new, among many, initialization strategies, activation functions (e.g., ReLu [14]) and adaptive optimization algorithms (e.g., Adam [15]). In the beginning, state-of-the-art was achieved in ASR by just replacing the GMMs with deep feedforward neural networks (DNNs), since the latter is better in modelling data that lie on or near a non-linear manifold, as opposed to the former [16].

These conventional ASR architectures, mentioned above, have limitations, mainly because they are based on HMMs and contain various sub-models that deal with separate acoustic, pronunciation and language models. First, the creation of all different models require expert knowledge and are time-consuming. They are also trained with different goals from the final evaluation metric, e.g., word error rate (WER). To create a state-of-the-art HMM-DNN system, it is first required to train an HMM-GMM system to obtain phonetic alignments. Moreover, since the decoding stage is performed by integrating all modules with finite-state transducers (FSTs), creating and implementing these well-optimized transducers is very complex. Finally, the HMM systems and n-gram language models make conditional independence assumptions, whereas real speech does not follow those strict assumptions.

A demanding task that is still ongoing is to fully replace this module-based architecture and replace the entire traditional pipeline with a fully differentiable DNN architecture to eliminate the above-outlined issues. This is possible, since deep learning technology enables the machine to create more abstract concepts out of simpler ones, as mentioned in [17]. Today, the main drawback of this technology, named end-to-end, is that it requires much more data (usually more than a thousand hours) and more computational power [2, 18], when compared to HMM-based systems. Notwithstanding, in the presence of big data, these systems achieve state-of-the-art results compared to HMM-based systems. Consequently, most modern ASR products provided by big companies like Amazon, Google and Apple, are based on end-to-end.

In 2006, it was proposed CTC, the first technique that was closer to an end-to-end system. CTC allows training end-to-end systems without requiring alignments between input features and output labels. It is still not an end-to-end system because CTC does not model the interdependencies between the outputs, which then requires a language model. To solve this CTC assumption problem, recurrent neural network (RNN)-transducer, or RNN-T, was proposed in 2012, where the CTC is jointly trained with an RNN that learns linguistic information. The other main technique for training end-to-end ASR is the attention end-to-end encoder-decoder system. This system is divided into an encoder, which acts as the acoustic model, an attention layer that learns the alignments between input and output. At last, the decoder acts as the language model. Usually, the outputs of these systems are characters or subwords. Beyond these main architectures for end-to-end ASR systems, there are many ways to create variants of it, by using adversarial training [19] and memory-based approaches [20, 21].

3. European Portuguese corpus

TRIBUS corpus training set is a collection of three training sets from three datasets: a read speech corpus, a broadcast news corpus and a telephone speech corpus, from INESC-ID. The validation and test sets of the TRIBUS corpus are the original ones from each corpus, except for SPEECHDAT [22], where the design process will be detailed below.

The read speech corpus used is BD-PÚBLICO [23]. Similar to Wall Street Journal corpus [24], BD-PÚBLICO was created from the Portuguese newspaper Público. Following, ALERT [25] is the European Portuguese broadcast news (BN) corpus, which contains spontaneous speech. ALERT, was created in cooperation with RTP, a public service broadcasting organization from Portugal. At last, there is SPEECHDAT corpus, a collection of speech read from telephone calls, collected by Portugal Telecom, a Portuguese telecommunications operator named Altice Portugal. Later, this collected data was designed and post-processed by INESC-ID. SPEECHDAT contains two main recording phases: SPEECHDAT 0 with 1000 speakers involved and the second, SPEECHDAT 1, with 4000 speakers involved. Each telephone call included in the database contains 33 red items and 7 spontaneous answers, where some contain demographic information. Only 9 phonetically rich sentences, from the set of 33 items, were used to create SPEECHDAT corpus for the ASR task.

As opposite to ALERT and BD-PÚBLICO, an experimental setup for SPEECHDAT was created. When working with SPEECHDAT, we noticed that from all the 36243 utterances from SPEECHDAT 1 only 3622 are unique, and from the total 9000 utterances from SPEECHDAT 0 only 904 are unique. Furthermore, SPEECHDAT 0 and SPEECHDAT 1 are two disjoint sets. For this reason, SPEECHDAT 1 was chosen for the

training set, and we divided SPEECHDAT 0 into two parts: the validation set and the test set. This data splitting process was made such that the number of female and male speakers was approximately the same for each set.

Table 1: *Summary of the corpora used to create the training set of the TRIBUS corpus.*

Corpus	Speech type	Hours
ALERT	broadcast news	60
BD-PÚBLICO	read	23
SPEECHDAT	telephone	63
Total		146

The TRIBUS corpus, depicted in Table 1, contains 146 hours and a total of 92184 labelled utterances for the training set. The validation and test sets for the TRIBUS corpus are presented in Table 2 and Table 3, respectively.

Table 2: *Summary of the TRIBUS corpus validation sets.*

Corpus	Hours
ALERT	8
BD-PÚBLICO	2
SPEECHDAT	9

Table 3: *Summary of the TRIBUS corpus test sets.*

Corpus	Hours
ALERT	6
BD-PÚBLICO	2
SPEECHDAT	9

The language model (LM) used for each set, when creating the HMM-based baseline, is the one that comes with each corpus, except for SPEECHDAT. To create the language model for SPEECHDAT, we first estimated a backoff 3-gram model with Kneser-Ney smoothing combined with Good-Turing smoothing. Since this LM is a flawed model to use for HMM-based ASR, due to the small linguistic variability in the training set, mentioned above, we interpolated this 3-gram LM model with BD-PÚBLICO LM [23]. The best combination of weights was 0.2 to the LM of SPEECHDAT and 0.8 to the LM of BD-PÚBLICO. An additional step was performed to normalize the notation of all the noise (e.g., _nsnoise_) and disfluencies (e.g., _ehm_hmm_) across the three datasets. Finally, for the TRIBUS corpus, we collected a lexicon of 108358 pronunciations, obtained from publicly available resources and all data was downsampled to 8kHz.

4. End-to-end model for European Portuguese ASR

Our end-to-end ASR system named TRIBUS, depicted in Figure 1, is a hybrid automatic speech recognizer that combines CTC with an attention network, which learns to map acoustic feature vectors to characters. The architecture is based on [26].

First, we will describe how the acoustic features are created. Next, the main idea behind the attention architecture used will be mentioned, with a detailed explanation for each module: *encoder*, *hybrid attention mechanism* and *decoder*. Finally, the end-to-end hybrid CTC-attention system will be described.

4.1. Acoustic features

The acoustic features consist in 80-dimensional Mel filterbank energies with pitch features, extracted with Kaldi [27], making the final size of the acoustic vector 83.

4.2. Attention-based architecture

The attention architecture contains three models: the encoder, the hybrid attention mechanism and decoder.

The encoder network

$$\boldsymbol{h}_t^{enc} = \mathit{Encoder}(\boldsymbol{x}), \tag{1}$$

converts the input features $\boldsymbol{x}$ into a framewise hidden vector $\boldsymbol{h}_t^{enc}$. Then, we have the hybrid attention weight computed as

$$\alpha_{ut} = \mathit{Hybrid\ attention}(\boldsymbol{q}_{u-1}, \{\alpha_{u-1}\}_{t=1}^{T}, \boldsymbol{h}_t^{enc}), \tag{2}$$

where α_{ut} is the weight that says how much attention is going to vector $\boldsymbol{h}_t^{enc}$, in order to compute output y_u, and $\boldsymbol{q}_{u-1}$ is the last hidden state of the long short-term memory (LSTM) [28] present in the decoder network, mentioned with more detail below. After computing all weights corresponding to all framewise hidden vectors $\boldsymbol{h}_t^{enc}$, we compute a weighted summation of hidden vectors $\boldsymbol{h}_t^{enc}$ to form the hidden vector $\boldsymbol{c}_u$,

$$\boldsymbol{c}_u = \sum_{t=1}^{T} \alpha_{ut} \boldsymbol{h}_t^{enc}. \tag{3}$$

At last, the decoder uses the weighted summation $\boldsymbol{c}_u$ and the last output y_{u-1} to compute the new output y_u:

$$p(y_u|y_1 ... y_{u-1}, \boldsymbol{x}) = \mathit{Decoder}(\boldsymbol{c}_u, y_{u-1}). \tag{4}$$

We will explain each module in more detail below.

4.2.1. Encoder network

Equation 1 converts the acoustic input features into a framewise vector $\boldsymbol{h}_t^{enc}$. The encoder network used consists of two initial blocks of the VGG layer [29], which yields better results most of the times than the pyramidal bidirectional long short-term memory (BLSTM) [26] architecture, proposed in [3]. With this, the number of frames is reduced approximately by a factor of 4. Following, there are 4 BLSTM layers with 1024 hidden and output units. Each BLSTM layer is followed by a linear projection layer, which receives 2048 features from the BLSTM layer and outputs 1024 features so that they can go to the next BLSTM layer. The final output is 1024 features for every reduced frame.

4.2.2. Hybrid attention mechanism

Hybrid attention mechanism, in equation 2, is decomposed as:

$$\{\boldsymbol{f}_t\}_{t=1}^{T} = \boldsymbol{K} * \boldsymbol{\alpha}_{u-1}, \tag{5}$$

where each $\boldsymbol{f}_t$ is a vector of size 10. $*$ denotes a 1D convolution operation along axis t, with the convolution parameter $\boldsymbol{K}$, to produce the set of features $\{\boldsymbol{f}_t\}_{t=1}^{T}$.

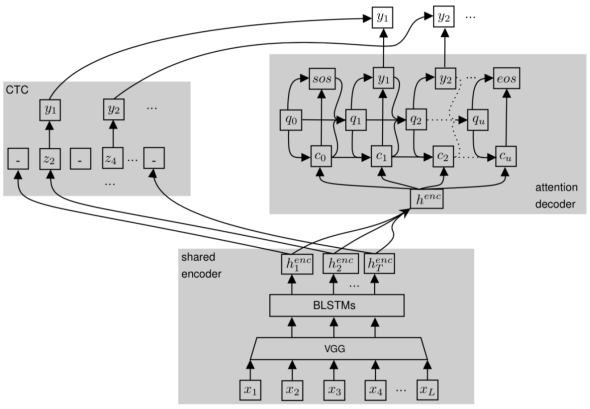

Figure 1: *TRIBUS hybrid CTC-attention architecture.*

Then, we can compute the energy value as:

$$e_{ut} = \boldsymbol{g}^T tanh(\text{LinearNN}(\boldsymbol{q}_{u-1}) + \text{LinearNNB}(\boldsymbol{h}_t^{enc}) + \text{LinearNN}(\boldsymbol{f}_t)), \quad (6)$$

where LinearNN is a linear layer with learnable matrix parameters and LinearNNB is a linear layer with learnable matrix and bias vector parameters. The number of output features used for the three linear networks is 320. Then, we are going to use all e_{ut} values and apply a softmax function to get the attention weight α_{ut}, so that we can compute the target output y_u:

$$\alpha_{ut} = Softmax(\{e_{ut}\}_{t=1}^{T}). \quad (7)$$

4.2.3. Decoder network

The decoder network, equation 4, is another recurrent neural network (RNN) that computes:

$$Decoder(.) = Softmax(\text{LinearNNB}(LSTM_u)). \quad (8)$$

The $LSTM_u$ is conditioned on three variables:

1. the previous hidden state $\boldsymbol{q}_{u-1}$;
2. the ground truth character, y_{u-1}, which is extracted from an embedding layer, trained while training the full end-to-end network;
3. the attention vector $\boldsymbol{c}_u$, which is concatenated with the previous character vector, giving a vector of size 2048 as input to the $LSTM_u$ cell;

For this architecture, two LSTM cells with 1024 units were used. The new hidden state $\boldsymbol{q}_u$ is computed as:

$$\boldsymbol{q}_u = LSTM_u(\boldsymbol{q}_{u-1}, \boldsymbol{c}_u, y_{u-1}). \quad (9)$$

4.3. Hybrid CTC-attention network

After describing the attention-based architecture, we can detail how the hybrid CTC-attention architecture works. The main idea is that the CTC and attention decoder networks share the same encoder, mentioned above. Also, when training, the CTC and attention loss are combined, to achieve more robustness and converge faster [8]:

$$Loss_{Total} = \lambda Loss_{CTC} + (1-\lambda) Loss_{Attention}, \quad (10)$$

where $\lambda \in [0, 1]$. Next, all noise and disfluencies from the TRIBUS corpus mentioned in Section 2 are mapped to a special token named <noise>. Also, it is important to note that there are special tokens for CTC and attention-based systems among all other output characters that exist, respectively. CTC requires a <blank> token [1], and the attention architectures requires the *start-of-sentence* and *end-of-sentence* (<sos/eos>) token. Therefore, the full hybrid CTC-attention system will have two special tokens plus an unknown token, <unk>, to map out-of-vocabulary (OOV) symbols. Finally, the total number of output symbols for TRIBUS is 49.

5. End-to-end speaker invariant

In order to make the ASR end-to-end system more invariant to the speaker, three experiments were performed. However, before proceeding, it is relevant to note that the first two experiments were preliminary experiments with SPEECHDAT corpus, and the last experiment was performed using ALERT corpus. The baseline architecture for all experiments is the same as mentioned in Section 3 for TRIBUS.

The first experience consisted of appending speaker iVectors to each acoustic feature vector, for the training, validation and test set, with the auxiliary of Kaldi toolkit. The second experience, similar to the first one, used iVectors extracted for each utterance instead of speaker iVectors, and at last, for the third experience, instead of depending on the extraction of more embedding vectors, a variation of adversarial training was applied. Gradient reversal, as proposed in [30], was used for the adversarial training. The main goal was to create a new network, detailed below, to classify the respective speaker ID. When training, this new network that shares the encoder with the hybrid CTC-attention decoder maximizes the probability of the utterance corresponding to the ground truth speaker. For the back-propagation, when arriving before the encoder, the gradient is reversed by a small factor, between 0 and 1. Thus, the encoder will be trained to create representations more invariant to the speaker ID. As a consequence, in principle, this helps the end-to-end ASR model to generalize better for unseen speakers on the test set.

The new classifier network architecture receives as input the vector of dimension 1024, which is an average of all vectors outputted by the encoder. A feedforward layer then transforms the averaged vector to a dimension of size 512, followed by a ReLU layer. Next, a second feedforward layer transforms the 512-dimensional vector into the same dimension, followed by another ReLU. At last, a feedforward layer transforms the 512-dimensional vector into a vector of dimension of size 1366, which is the number of speakers that exist on the ALERT training set. The network is then trained to classify the respective ground truth speaker. The factor, α, used for the reversal of the gradient, mentioned above, was chosen to be computed as:

$$\alpha = \frac{2}{1 + e^{-0.07 * epoch}} - 1, \quad (11)$$

where $epoch$ is the current epoch of the training stage. It is important to remind that this network is removed when applying the hybrid CTC-attention system to the validation and test sets.

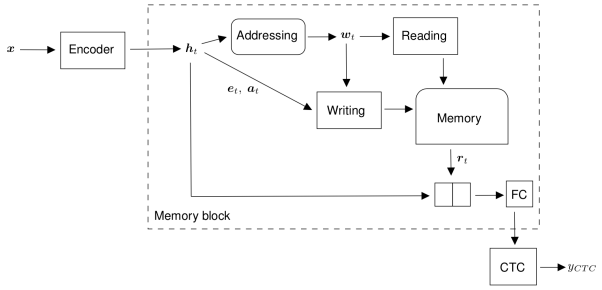

Figure 2: *Block diagram of the CTC ASR system with a memory-based adaptation network.*

6. CTC-based memory network

Neural Turing machines (NTM) [31] despite having been created in 2014, were only applied to a small number of tasks until nowadays, mainly because of their difficulty to train [32]. For end-to-end ASR only two known applications, from 2020, use NTMs. The first one is from [20], where the NTM is used to store iVectors and read from them to combine with the hidden vectors of the encoder. In this work, it is essential to note that no write operation for the NTM exists, therefore this approach ignores the full potential of the NTMs. The other known work is [21], where they combine the NTM with the decoder network, in order to improve the LM of the end-to-end model. Conversely to the first-mentioned approach that uses the NTM, our experiences use the write head. Also, they do not include any extracted iVectors. All experiences we created for the NTM use ALERT corpus with the baseline architecture mentioned in Section 3.

In the beginning, we started by combining the NTM with the hybrid CTC-attention architecture. It was shortly discovered that the improvements were small when tuning the NTM hyperparameters, but when giving more weight to the CTC module, the improvements started to increase gradually. Consequently, all future experiments that were performed only use the CTC module, i.e., the λ parameter from equation 10 is set to 1. Now, we will look at how the NTM was combined with the CTC-based network. Before proceeding to the description of the memory-based architecture, it is crucial to notice that all algorithmic details about the NTM are from [32].

The architecture for the CTC ASR memory-based system is depicted in Figure 2. For each hidden vector, $\boldsymbol{h}_t$, emitted by the encoder, the Addressing block from Figure 2 will follow all required steps, detailed in [32], to create the weights $\boldsymbol{w}_t$. At the same time, the erase vector, $\boldsymbol{e}_t$, and add vector, $\boldsymbol{a}_t$, are also extracted from $\boldsymbol{h}_t$. From this point, the memory starts to read from and write to, following the detailed processes mentioned in [31]. After the read and write updates, the read vector, $\boldsymbol{r}_t$, is concatenated with the hidden vector, $\boldsymbol{h}_t$. Next, this concatenated vector goes trough a fully connected (FC) layer, which reduces the vector dimension to the original size, i.e., 1024. This procedure is sequential for all vectors, $\boldsymbol{h}_t$, created by the encoder. Therefore, there is no significant increase of delay for the CTC-based system, as opposed to the delay created when the attention module waits for the encoder to finish all computations.

7. Experiments

For all end-to-end experiments, we used the second version of ESPnet toolkit [26] to implement and investigate our proposed methods, which is still under development by the time we write this document. First, to evaluate our end-to-end ASR TRIBUS system performance, we compare it to a robust HMM-DNN baseline using the same corpus. Finally, in this section, we also present the experiments for the speaker invariant and CTC ASR memory-based approach.

7.1. End-to-end experiments

For the end-to-end training, it was used a single GeForce GTX 1080 Ti. It was also used a learning rate of 1.0, and the model was trained for 30 epochs, with early stopping (patience of 4 epochs). Adadelta [33], an adaptive learning rate back-propagation algorithm, was the optimizer chosen, with a batch size of 30 and gradient clipping of 5. All weights were initialized using Xavier initialization [34]. It was also used a scheduler for the learning rate, where the scale factor was 0.5 and the patience 1 epoch. For data augmentation, we used speed perturbed factors of 0.9, 1.0 and 1.1, and SpecAugment [35]. The decoding process of the hybrid CTC-attention model follows the setup in [26]. It is relevant to note that no language model was used to train the end-to-end ASR system.

The WER results for the end-to-end TRIBUS ASR system, are presented in Table 4. From the results, we can see that BD-PÚBLICO has the lowest WER, mainly because it is read speech.

Table 4: *WERs [%] on the end-to-end ASR validation (valid) and test sets of the TRIBUS corpus.*

	valid	**test**
ALERT	18.80	19.40
BD-PÚBLICO	8.60	9.10
SPEECHDAT	21.20	20.00

7.2. HMM-based experiments

To create a robust HMM-based baseline for the TRIBUS corpus, we designed a similar procedure to the 's5' recipe of WSJ corpus, from Kaldi. First, to create the alignments for the HMM-DNN system, we trained an HMM-GMM system using the TRIBUS corpus, mentioned in Section 2. The training stages that created the HMM-GMM system are the following: (1) monophone stage, (2) triphones + delta + delta-delta stage, (3) triphones + LDA + MLLT stage and finally, (4) the triphones + SAT stage. For the first training stage (1), only the 2000 shortest utterances from the training set were used. For the second (2), a subset of 30000 utterances from the total of 92184, mentioned in Section 3, were used. Finally, for the last two training stages, (3) and (4), all utterances were used. After creating the HMM-GMM system, the HMM-DNN system was trained following the Chain recipe from WSJ ("run_tdnn_1i.sh"), in Kaldi. The main difference is that only 12 layers were used to train the model, instead of 13 layers.

The WER results for the HMM-DNN ASR system, with respect to the TRIBUS corpus, are presented in Table 5. From results, we can notice that BD-PÚBLICO achieves the lowest WER. When comparing with the WERs from Table 4, we can observe that there is still a significant difference between the

Table 5: *WERs [%] on the HMM-based validation (val) and test sets of the TRIBUS corpus.*

	valid	test
ALERT	9.69	9.65
BD-PÚBLICO	2.56	3.04
SPEECHDAT	2.49	4.86

traditional HMM-DNN ASR systems and the end-to-end ASR systems, for low resources.

Finally, it is essential to notice that our baseline system created a new state-of-the-art result for ALERT compared to the original work [25]. Using the same training set, test set and language model, we were able to decrease the absolute WER from 23.50% to 9.65%.

7.3. HMM-based vs end-to-end European Portuguese systems

For a fair comparison between the end-to-end ASR model and the HMM-DNN ASR model evaluation using CER was performed in the HMM-DNN test sets. The CER results for the end-to-end TRIBUS model and the HMM-DNN baseline system, are depicted in Table 6.

Table 6: *CERs [%] on the end-to-end ASR TRIBUS system and HMM-DNN ASR system, on test sets.*

	TRIBUS	HMM-DNN
ALERT	8.40	4.33
BD-PÚBLICO	2.70	0.95
SPEECHDAT	8.40	3.26

By observing the results from Table 6, we notice that the CERs of the proposed ASR system, TRIBUS, are very close to CERs of the HMM-DNN baseline system, as opposed to the WERs. In particular, for BD-PÚBLICO, the CER from TRIBUS system, 2.70%, is very close to the CER of the HMM-DNN system, 0.95%, where the absolute difference between the two is only 1.75%. Nevertheless, the HMM-based systems are still better than the end-to-end systems, in low resource settings, according to the CER metric.

7.4. Speaker invariant experiments

In the first part, we will present the experimental results that used iVectors to create speaker invariant end-to-end ASR systems. In the second part, the experiments that used adversarial training for the same purpose will be shown.

7.4.1. Speaker and utterance iVectors results

The WERs and CERs results for the speaker iVectors, utterance iVectors and for the preliminary baseline of SPEECHDAT (an hybrid CTC-attention system), are presented in Table 7 and Table 8, respectively.

We can observe that neither speaker iVectors nor utterance iVectors perform better than the original baseline from the WER results. However, we can see for the CER test set performance that the utterance iVectors achieves 4.90% while the baseline only obtains 5.00% CER. Despite this result, appending speaker or utterance iVectors to the initial acoustic vectors do not significantly improve hybrid CTC-attention systems performance.

Table 7: *WERs [%] for SPEECHDAT validation and test sets - iVectors.*

	valid	test
Baseline	8.80	8.90
Baseline + speaker iVectors	9.60	9.70
Baseline + utterance iVectors	9.20	9.00

Table 8: *CERs [%] for SPEECHDAT validation and test sets - iVectors.*

	valid	test
Baseline	5.00	5.00
Baseline + speaker iVectors	5.10	5.20
Baseline + utterance iVectors	5.00	4.90

7.4.2. Adversarial training results

The test set WERs and CERs results for the adversarial training and baseline of ALERT (an hybrid CTC-attention system), are presented in Table 9.

Table 9: *WERs [%] and CERs [%] for ALERT test set corpus - adversarial training.*

	valid	test
Baseline	20.60	8.40
Baseline + adversarial training	23.20	9.70

From results, we can see that this kind of adversarial training does not improve when compared to the baseline hybrid CTC-attention system. After training several times the adversarial training system, to achieve the shown results, it was discovered that "deep models already learn speaker-invariant representations" in the work of [19]. The mentioned work did a similar experience for CTC-based models and found out that adversarial training for speaker invariant did not improve the baseline architecture. Finally, we can conclude that, with enough layers, the encoder of the end-to-end ASR system learns already invariant speaker representations.

7.5. CTC-based memory network experiments

All WERs and CERs results are presented in Table 10 and Table 11, respectively. The NTM baseline architecture, which is only denoted by NTM in both tables, contains only 1 head for reading and writing, 1 of shift, 128 rows by 20 columns (128x20) and all memory entries are initialized with zeros. The information that appears between parentheses shows what is changed over the NTM baseline, mentioned above.

We can see from Table 10 that the CTC baseline plus the memory-based approach, using 128 rows and 40 columns, achieves the best WER on the test set, 33.60%. More importantly, we notice that the WERs of the CTC memory-based approaches always improve over the CTC baseline. For example, the CTC-based model that uses the NTM configured with 128 rows and 40 columns, decreases the WER from 35.30% to

Table 10: *WERs [%] for ALERT corpus - NTM.*

	valid	test
CTC baseline	35.00	35.30
CTC baseline + NTM	34.10	35.10
CTC baseline + NTM (225x20)	33.60	33.80
CTC baseline + NTM (128x40)	**33.00**	**33.60**
CTC baseline + NTM (225x40)	34.00	34.50
CTC baseline + NTM (2 heads)	33.10	33.70

Table 11: *CERs [%] for ALERT corpus - NTM.*

	valid	test
CTC baseline	11.70	11.80
CTC baseline + NTM	11.50	11.70
CTC baseline + NTM (225x20)	11.30	11.30
CTC baseline + NTM (128x40)	**11.20**	**11.20**
CTC baseline + NTM (225x40)	11.40	11.60
CTC baseline + NTM (2 heads)	**11.20**	**11.20**

33.60% on the test set. An absolute WER difference of 1.70%. Additionally, when observing the table with the CERs, we see that there are two models with the lowest CER. One of them is the one that achieved the best WER, mentioned above, and the other is a CTC with an NTM that uses 2 heads for reading and writing, instead of only one. Nevertheless, the best model is the CTC with an NTM that contains a memory of size 128x40.

Overall, in principle, CTC-based systems can improve the WER and CER performance if combined with an NTM. In the near future, more configurations will be tried out, and all experiments presented in Table 10 and Table 11 will be replicated for the WSJ and Librispeech corpora, since, up to the best of our knowledge, there is no work done in this way.

8. Conclusions

In this document, we presented the first known work using state-of-the-art end-to-end systems for low resource European Portuguese. From the experimental results, we can see that the TRIBUS end-to-end WER performance is not very close to the HMM-based results, but is comparable according to the CERs. The HMM-DNN models still have the advantage of having a language model that comprises almost all linguistic variation present in validation and test sets, and a pronunciation dictionary as well. Nonetheless, it is already impressive that the end-to-end hybrid CTC-attention systems can learn so much without any language model or pronunciation dictionary. We also observed that appending iVectors to the input acoustic features or using adversarial training for speaker invariant, does not improve the performance of the end-to-end system. If they do, the improvements are not very significant. At last, and more notably, we proposed a novel way of training CTC-based models using a memory-based system inspired by the NTM model. The improvements, when compared to a normal CTC-based system, are quite remarkable.

9. Future work

At first, more configurations for the memory-based approach applied to end-to-end CTC-based ASR will be tried out forthwith. Simultaneously, due to the successful results for ALERT, the memory-based experiments will be replicated for English corpora and published for INTERSPEECH 2021, since, up to the best of our knowledge, there is no work done in this way.

More importantly and at last, this work provides the groundwork and suggests future work paths in end-to-end ASR with low resource settings, e.g., European Portuguese. Many important problems remain, but the main one is the *problem of low resources* within deep learning. Deep neural networks are known to require many labelled data in order to achieve state-of-the-art performance results. Nevertheless, it is known that gathering and labelling data is very time consuming and also very expensive. One way to solve this problem would be to approach the end-to-end ASR system with *unsupervised learning*. Unsupervised learning algorithms try to find relevant "structure" in data, instead of learning to perform a specific classification or regression task. Motivated by the fact that children learn how the world works by observation and remarkably little interaction, i.e., with little supervised feedback, unsupervised learning is a promising research path for end-to-end ASR with low resources. When approaching unsupervised learning in deep learning, one of the main problems that arise is how to find a good representation that could, in principle, disentangle all relevant factors for the ASR task. The problem of "discovering good representations" is approached with more detail in [36].

10. Acknowledgements

We want to thank INESC-ID for providing both the data and computational resources, that made this work possible.

11. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 369–376. [Online]. Available: https://doi.org/10.1145/1143844.1143891

[2] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1764–1772. [Online]. Available: http://proceedings.mlr.press/v32/graves14.html

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 4960–4964.

[4] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 577–585.

[5] A. Graves, "Sequence transduction with recurrent neural networks," 2012.

[6] A. Y. Hannun, A. L. Maas, D. Jurafsky, and A. Y. Ng, "First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns," 2014.

[7] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," 2014.

[8] S. Watanabe, T. Hori, S. Kim, J. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition,"

IEEE Journal on Selected Topics in Signal Processing, vol. 11, no. 8, pp. 1240–1253, Dec. 2017.

[9] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," 2014.

[10] A. Rousseau, P. Deléglise, and Y. Estève, "Ted-lium: an automatic speech recognition dedicated corpus." in *LREC*, N. Calzolari, K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, Eds. European Language Resources Association (ELRA), 2012, pp. 125–129. [Online]. Available: http://dblp.uni-trier.de/db/conf/lrec/lrec2012.html#RousseauDE12

[11] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, 2015.

[12] A. Ng, *Machine Learning Yearning*. Online Draft, 2017.

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, p. 1527–1554, Jul. 2006. [Online]. Available: https://doi.org/10.1162/neco.2006.18.7.1527

[14] R. Hahnloser, R. Sarpeshkar, M. Mahowald, and R. Douglas, "Digital selection and analog amplification co-exist in an electronic circuit inspired by neocortex," *Nature*, 01 2000.

[15] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[16] G. Hinton, l. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, pp. 82–97, 11 2012.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[18] Y. Zhang, W. Chan, and N. Jaitly, "Very Deep Convolutional Networks for End-to-End Speech Recognition," *arXiv:1610.03022 [cs]*, Oct. 2016, arXiv: 1610.03022. [Online]. Available: http://arxiv.org/abs/1610.03022

[19] Y. Adi, N. Zeghidour, R. Collobert, N. Usunier, V. Liptchinsky, and G. Synnaeve, "To reverse the gradient or not: An empirical comparison of adversarial and multi-task learning in speech recognition," *CoRR*, vol. abs/1812.03483, 2018. [Online]. Available: http://arxiv.org/abs/1812.03483

[20] L. Sarı, N. Moritz, T. Hori, and J. L. Roux, "Unsupervised speaker adaptation using attention-based speaker memory for end-to-end asr," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7384–7388.

[21] D. Lee, J.-S. Park, M.-W. Koo, and J.-H. Kim, "Language model using ring machine based on localized content-based addressing," *Applied Sciences*, vol. 10, no. 20, p. 7181, 2020.

[22] A. Hagen and J. Neto, "Hmm/mlp hybrid speech recognizer for the portuguese telephone speechdat corpus," 06 2003, pp. 126–134.

[23] J. Neto, C. Martins, H. Meinedo, and L. Almeida, "The design of a large vocabulary speech corpus for portuguese." 01 1997.

[24] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, Banff, 1992, pp. 899–902.

[25] H. Meinedo, D. Caseiro, J. Neto, and I. Trancoso, "Audimus.(media): A broadcast news speech recognition system for the european portuguese language," 06 2003, pp. 9–17.

[26] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "Espnet: End-to-end speech processing toolkit," 2018.

[27] D. Povey, A. Ghoshal, G. Boulianne, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, and G. Stemmer, "The kaldi speech recognition toolkit," in *In IEEE 2011 workshop*, 2011.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[30] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," 2016.

[31] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *CoRR*, vol. abs/1410.5401, 2014. [Online]. Available: http://arxiv.org/abs/1410.5401

[32] M. Collier and J. Beel, "Implementing neural turing machines," 2018.

[33] M. D. Zeiler, "Adadelta: An adaptive learning rate method," 2012.

[34] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.

[35] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Interspeech 2019*, Sep 2019. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680

[36] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," 2014.